ADVANCE JAVA

LAB-1

Name: Rajeev Raj

Enroll: A4530482102<u>5</u>

Submitted to : Dr. Naveen Kumar Singh

CRUD OPERATION

Problem Description :

Creating a Note taking app which allow user to create notes and store it. This should achieved using java programming language and for storing data use mySQL database. To connect java with mySQL we use Java Database Connectivity(JDBC).

Operations to Achieve :

→ (C) CREATE DATA : allow user to insert new data
→ (R) RETRIEVE DATA : allow user to retrieve/read data
→ (U) UPDATE DATA : allow user to update data
→ (D) DELETE DATA: allow user to delete data

DESIGN

The design of the problem statement for creating a simple Java application that establishes JDBC connection and performs CRUD operations involves several key components and considerations:

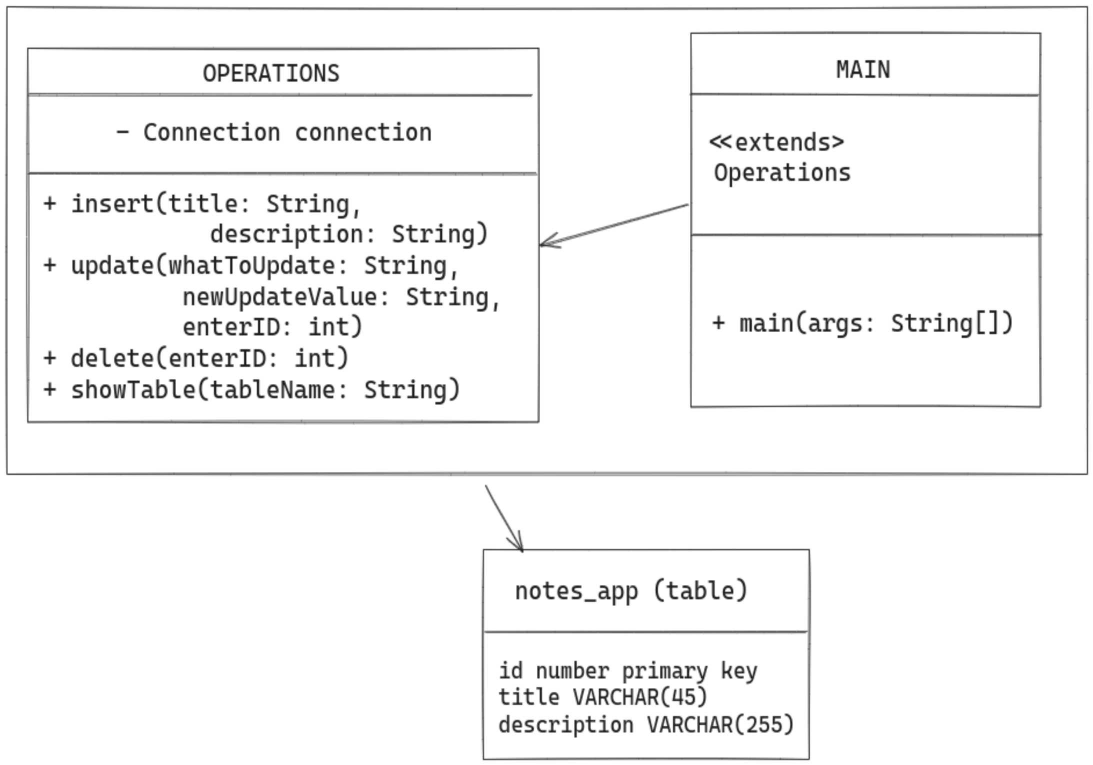
1. **User Interface Design :** Upon running the application, users will be presented with a menu containing 5 options, with 4 of them representing crud operations and the last option for exiting the application gracefully. Based on the user's choice, the application will invoke the appropriate method from the Student class to perform the CRUD operation.

2. **Database Connection Management:** The application needs to establish a JDBC connection with the relational database system using the correct connection details.

3. **Error Handling:** Error handling should be implemented to manage exceptions during database operations.

4. **Code Modularity and Maintainability:** The application's code should be modular and well-organized, following best practices in software design and development. It should be easy to maintain and extend, allowing for future enhancements or modifications without significant refactoring.

5. **Class Diagram:** A class diagram is crucial for design purposes as it visually illustrates the structure, relationships, and behavior of classes within a system. It aids in organizing and conceptualizing software components, facilitating communication among developers, guiding implementation, and ensuring consistency and scalability throughout the design process. Here's a class diagram demonstrating our problem statement



CODE :-

OPERATIONS.JAVA

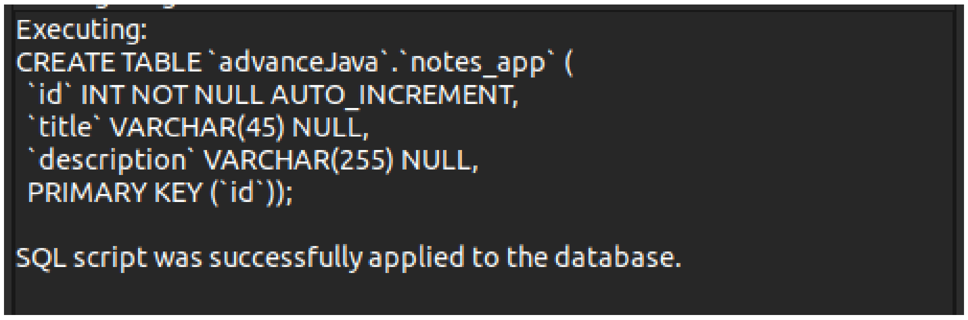
```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.PreparedStatement;

public class Operations {
    public static void main(String[] args) {

    }
    static void update(String whatToUpdate , String newUpdateValue , int enterID){
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/advanceJava", "root",
"rajeevIsGod");


            String updateQuery = "UPDATE notes_app SET " + whatToUpdate + " = ? WHERE id =
?";
            PreparedStatement preparedStatement = con.prepareStatement(updateQuery);


            preparedStatement.setString(1, newUpdateValue);
            preparedStatement.setInt(2, enterID);


            int rowAffected = preparedStatement.executeUpdate();

            System.out.println(rowAffected + " rows affected");

            preparedStatement.close();
            con.close();
        } catch (Exception e) {
            System.out.println("Error: " + e);
        }
    }
    static void insert(String title , String description){
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/advanceJava", "root",
"rajeevIsGod");

            // Use PreparedStatement to create a parameterized query
            String insertQuery = "INSERT INTO notes_app (title, description) VALUES (?, ?)";
            PreparedStatement preparedStatement = con.prepareStatement(insertQuery);

            // Set the values for the placeholders
            preparedStatement.setString(1, title);
            preparedStatement.setString(2, description);

            // Execute the insert
            int rowAffected = preparedStatement.executeUpdate();

            System.out.println(rowAffected + " rows affected");

            preparedStatement.close();
            con.close();
        } catch (Exception e) {
            System.out.println("Error: " + e);
        }
    }

    static void delete(int enterID){
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/advanceJava", "root",
"rajeevIsGod");
            Statement st = con.createStatement();

            // Use the parameterized query to delete a specific record by ID
            int rowAffected = st.executeUpdate("DELETE FROM notes_app WHERE id = " +
enterID);

            System.out.println(rowAffected + " rows affected");

            st.close();
            con.close();
        } catch (Exception e) {
            System.out.println("Error: " + e);
        }
    }
    static void showTable() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/advanceJava", "root",
"rajeevIsGod");
            Statement st = con.createStatement();
            ResultSet resultSet = st.executeQuery("SELECT * FROM notes_app" );

            while (resultSet.next()) {
                int id = resultSet.getInt("id");
                String name = resultSet.getString("title");
                String address = resultSet.getString("description");

                System.out.println("\n ID: " + id + "\n Title: " + name + "\n Address: " + address);
                System.out.println();
                System.out.println();
            }

            st.close();
            con.close();
        } catch (Exception e) {
            System.out.println("Error: " + e);
        }
    }
}
```

MAIN.JAVA

```
import java.util.Scanner;

public class Main extends Operations {
    public static void main(String[] args) {
        System.out.println("*********  Welcome to this note-making app ***********");
        Scanner sc = new Scanner(System.in);
        while (true) {
            System.out.println();
            System.out.println("1. INSERT");
            System.out.println("2. UPDATE");
            System.out.println("3. DELETE");
            System.out.println("4. SHOW TABLE");
            System.out.println("5. EXIT");
            System.out.println();
            int opt = sc.nextInt();
            sc.nextLine();

            if (opt == 1) {
                System.out.println("Enter Title");
                String title = sc.nextLine();
                System.out.println("Enter Description");
                String desc = sc.nextLine();
                insert(title, desc);
            } else if (opt == 2) {
                System.out.println("Enter what you want to update");
                String updateVal = sc.nextLine();

                System.out.println("Enter value");
                String value = sc.nextLine();
                System.out.println("Enter id");
                int id = sc.nextInt();
                update(updateVal, value, id);
            } else if (opt == 3) {
                System.out.println("Enter id whom you want to delete");
                int id = sc.nextInt();
                delete(id);
            } else if (opt == 4) {
                showTable();
            }else if(opt ==5){
                break;
            }
        }
    }
}
```

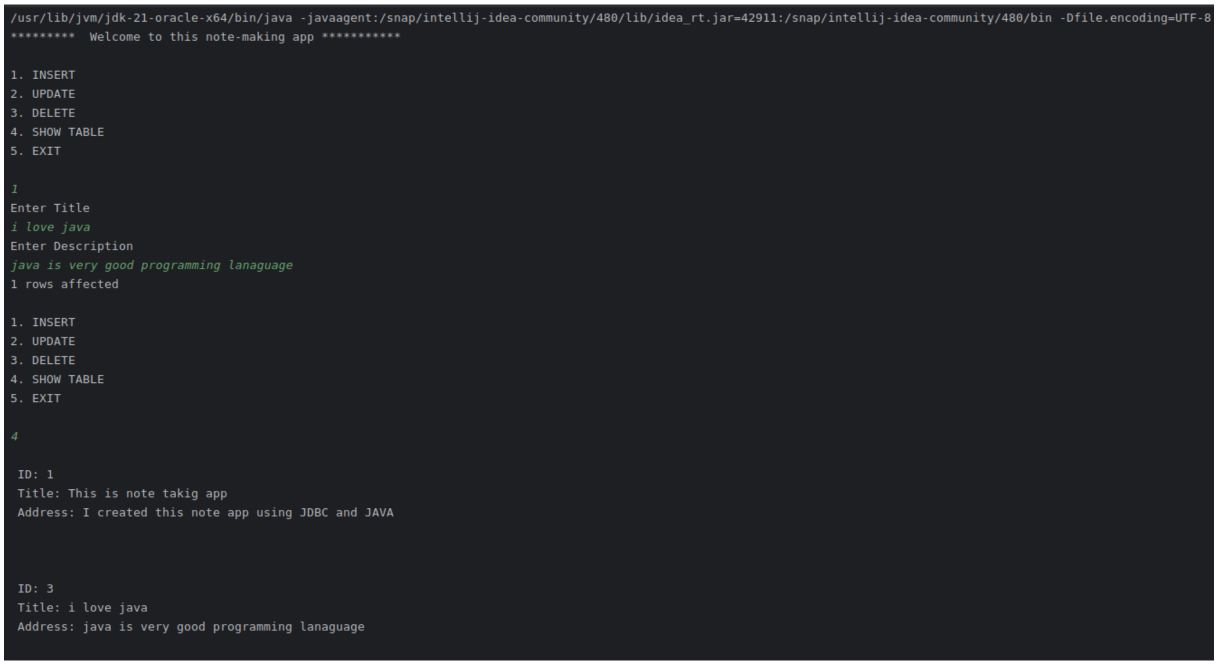
INPUT/OUTPUT

CREATE TABLE:



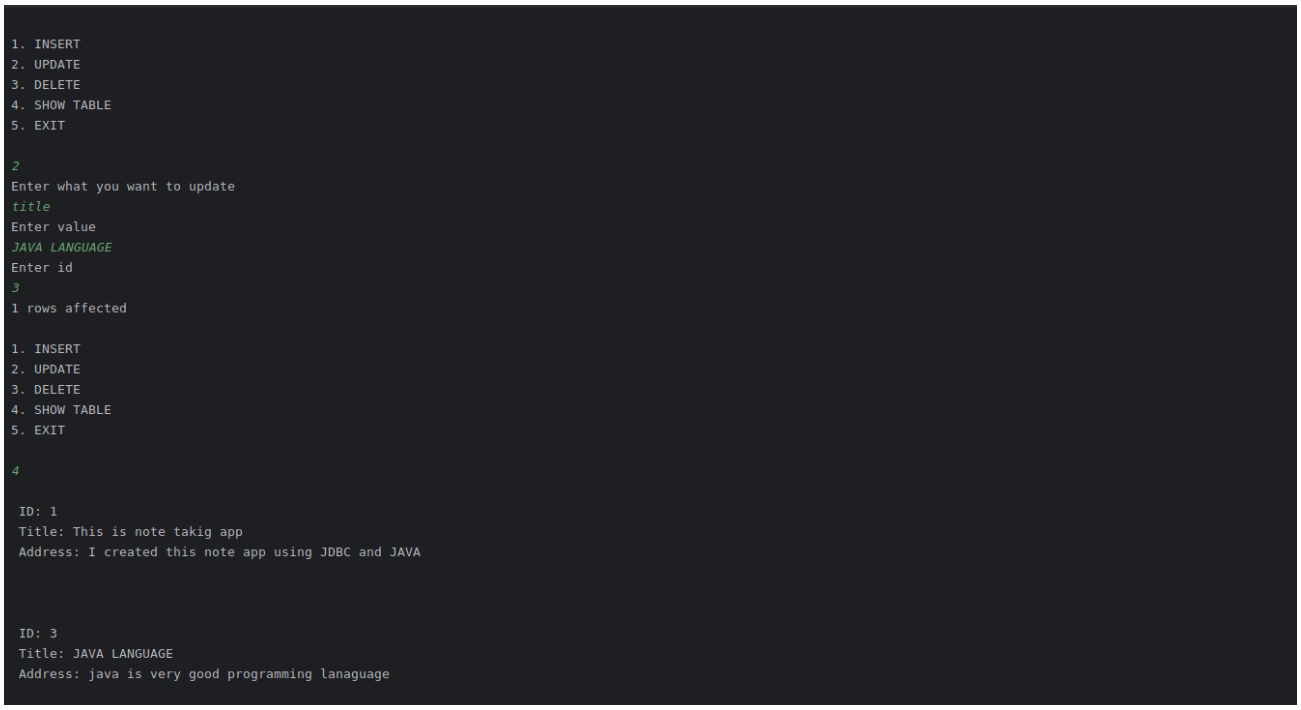
```
Executing:
CREATE TABLE `advanceJava`.`notes_app` (
 `id` INT NOT NULL AUTO_INCREMENT,
 `title` VARCHAR(45) NULL,
 `description` VARCHAR(255) NULL,
 PRIMARY KEY (`id`));

SQL script was successfully applied to the database.
```

INSERT DATA:



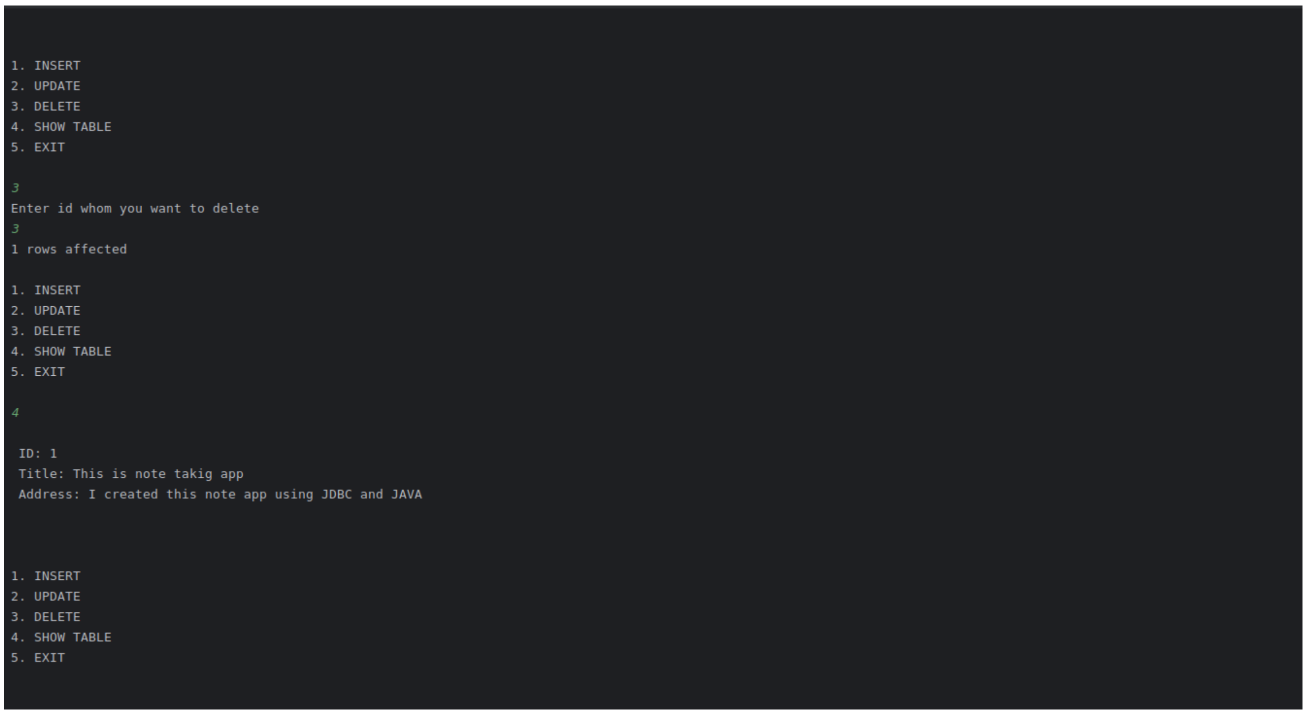
```
/usr/lib/jvm/jdk-21-oracle-x64/bin/java -javaagent:/snap/intellij-idea-community/480/lib/idea_rt.jar=42911:/snap/intellij-idea-community/480/bin -Dfile.encoding=UTF-8
*********  Welcome to this note-making app ***********

1. INSERT
2. UPDATE
3. DELETE
4. SHOW TABLE
5. EXIT

1
Enter Title
i love java
Enter Description
java is very good programming lanaguage
1 rows affected

1. INSERT
2. UPDATE
3. DELETE
4. SHOW TABLE
5. EXIT

4

 ID: 1
 Title: This is note takig app
 Address: I created this note app using JDBC and JAVA



 ID: 3
 Title: i love java
 Address: java is very good programming lanaguage
```

UPDATE DATA:



```
1. INSERT
2. UPDATE
3. DELETE
4. SHOW TABLE
5. EXIT

2
Enter what you want to update
title
Enter value
JAVA LANGUAGE
Enter id
3
1 rows affected

1. INSERT
2. UPDATE
3. DELETE
4. SHOW TABLE
5. EXIT

4

 ID: 1
 Title: This is note takig app
 Address: I created this note app using JDBC and JAVA



 ID: 3
 Title: JAVA LANGUAGE
 Address: java is very good programming lanaguage
```

DELETE DATA:



```
1. INSERT
2. UPDATE
3. DELETE
4. SHOW TABLE
5. EXIT

3
Enter id whom you want to delete
3
1 rows affected

1. INSERT
2. UPDATE
3. DELETE
4. SHOW TABLE
5. EXIT

4

 ID: 1
 Title: This is note takig app
 Address: I created this note app using JDBC and JAVA



1. INSERT
2. UPDATE
3. DELETE
4. SHOW TABLE
5. EXIT
```